N91-20691

CAPTURING, USING, AND MANAGING QUALITY ASSURANCE KNOWLEDGE FOR SHUTTLE POST-MECO FLIGHT DESIGN

H. L. Peters, L. R. Fussell, M. A. Goodwin
Rockwell Space Operations Company
600 Gemini
Houston, Texas 77058-2777
(713)282-2861

R. D. Schultz
Abacus Programming Corporation
14545 Victory Boulevard, Suite 300
Van Nuys, California 91411
(818)785-8000

ABSTRACT

Ascent initialization (I-load) values used by the Shuttle's onboard computers for nominal and abort mission scenarios are verified by a six degree-of-freedom computer simulation. The procedure that the Ascent Post Main Engine Cutoff (Post-MECO) group uses to perform quality assurance (QA) of the simulation is time consuming. Also, the QA information, checklists and associated rationale, though known by the group members, is not sufficiently documented, hindering transfer of knowledge and problem resolution. A new QA procedure which retains the current high level of integrity while reducing the time required to perform QA is needed to support the increasing Shuttle flight rate. Documenting the knowledge is also needed to increase its availability for training and problem resolution.

To meet these needs, a knowledge capture process, "embedded" into the group activities, has been initiated to verify the existing QA checks, define new ones and document all rationale. The resulting checks have been automated in a conventional software program to achieve the desired standardization, integrity and time reduction. A prototype electronic knowledge base has been developed with Macintosh's HyperCard™ to serve as a knowledge capture tool and a knowledge repository. It has also been designed to support a future capability to automatically generate code. The success of the effort has been demonstrated by the adaptation of the knowledge capture process by two other Ascent groups.

INTRODUCTION

This paper focuses on capturing quality assurance (QA) knowledge for the unique environment and needs that exist in the Flight Design and Dynamics Department (FD&DD) of the Rockwell Space Operations Company (RSOC). The quality assurance knowledge capture process began with a group within FD&DD that designs the post-MECO (Main-Engine Cutoff) phase of the space shuttle ascent trajectory. This post-MECO project is serving as the prototype and pathfinder for subsequent QA knowledge capture projects.

Faced with an increasing flight rate, FD&DD is looking for ways to increase productivity while containing the number of personnel. FD&DD consists of approximately 500 employees who design the ascent, orbit, and entry trajectories for shuttle missions. It produces approximately 500 products for each mission, performs related analysis and provides real-time support in the Mission Control Center. The current flight rate of about nine missions per year is expected to increase to twelve in 1992. QA within FD&DD currently consists of highly manual, repetitive, and time consuming tasks. Both the externally delivered products as well as tasks intermediate to generating products require QA. The development and use of knowledge tools is recognized to be one source of productivity improvements. Knowledge tools have potential not only to reduce task time, but also to reduce risk and increase quality in QA.

The following sections discuss in detail the significant aspects of the project to date as well as future plans. First, several problems uncovered early in the knowledge capture effort are discussed. Discussed second are the knowledge capture process that evolved and the results, the knowledge tools. Presented next is the knowledge-based management system (KBMS) along with its utility for on-going knowledge maintenance, and other applications of the KBMS knowledge. In conclusion, the self-sustaining nature of the knowledge capture methodology, e.g., the knowledge capture process and resulting knowledge-base tools, is presented along with plans to apply the methodology in other areas.

THE KNOWLEDGE CAPTURE PROBLEM

The post-MECO QA task was selected as the initial QA task to be automated in FD&DD because of the potential high manpower reduction, and the enthusiasm and support of key group members. Post-MECO QA centers around the Space Vehicle Dynamics Simulator (SVDS), a six degree-of-freedom simulation of shuttle flight dynamics. The Post-MECO group uses SVDS to perform validation of initialization variables (I-loads) loaded in the onboard software for each mission. SVDS executes nominal post-MECO and abort scenarios using the I-loads in the input runstream. Significant flight-related parameters are output at specific events during the simulation. Group members "QA" the simulation by first detecting errors revealed in the output, determining their cause, and then determining the correction, analogous to fault detection, isolation, and resolution (FDIR). Errors are detected by applying boundary-value constraints to the SVDS output. Knowledge of the problem and simulation software are then used to determine the cause and to provide a fix.

Initially, capture of the QA knowledge involved some significant concerns, the environment and condition of the knowledge. The QA information passed on to FD&DD at the beginning of the Space Transportation Systems Operations Contract (STSOC) consisted only of a skeleton checklist of boundary-valued constraints for error detection. Rationales for the checks and problem resolutions were not documented. Although some of the experienced personnel relocated at STSOC, most of the post-MECO domain specialists were in the process of acquiring expertise.

To train the domain specialists the skeleton checklist was used with a large verbal transfer of knowledge. When an inexperienced engineer needed to resolve a violation of the checklist, assistance was often necessary. This assistance was provided by an experienced group member, the expert, who had knowledge in that specific problem area. The engineer did not always learn the rationale behind a constraint until he/she had to resolve a violation. As the engineers obtained experience, they improved their checklists, and the individual checklists diverged. However, there could be checks on a list which the engineer did not fully understand. The engineer performed these checks by rote, and for these checks worked with a "black box".

Due to heavy workloads of the experts, they were not always available. If the experts were unavailable, another method of resolving a violation was to determine the rationale of the constraints from formal documents. The formal documents are texts on flight design, flight rules, vehicle constraints, groundrules and constraints, and flight software requirements. There is then the problem of determining which document to search, locating it, and finding the specific information within it. Therefore, capture of the rationales would require much research and time. The heavy workload of the engineers limited the time any one of them could spend on research.

THE KNOWLEDGE CAPTURE APPROACH

After identifying the condition and environment of the knowledge, the knowledge capture approach required would

- Capture the dispersed rationales and problems resolutions.
- Support the in-progress training of the domain specialist engineers.
- Fit within the discipline specialists time constraints.
- Support long-term, continuous capture.
- Easily incorporate changes in the QA knowledge as the group's novices become experts (Reference 1).

- Prevent any resulting automation aids from becoming a black box, i.e., support knowledge retention.
- Support and speed-up training of future new-hires, expected because of routine turnover and additional workload caused by the increasing flight rate.
- Reduce the workload of the available knowledge engineers to free them for pending projects.

It was decided that these objectives could best be met by making knowledge capture an integral part of the group's work process and, if possible, by making it self-sustaining.

Because of the experience level of the engineers and the uncertainty about the return on automating problem resolution, it was also decided to divide the knowledge capture into two phases. Phase I would capture the error detection criteria used in the QA of the simulation runs and the rationales behind them. Phase II would focus on the problem resolution knowledge, i.e., determining the cause of an error and deciding on a fix. The first part of the effort in Phase I would focus on obtaining a consolidated list of boundary-value constraints. Immediate benefit could be realized by the group by automating these checks. A straight-forward FORTRAN program would be written, and used during Phase I. The second part of Phase I would concentrate on capturing constraint rationale and refinement of error-detection constraints; the refined checks would be used to update the FORTRAN program.

One person would not be able to gather all the lost knowledge due to the dispersed nature of the knowledge and personal time constraints. To overcome this problem, a divide-and-conquer approach was taken. This approach required involvement of all members of the Post-MECO group. Existing group meetings were used to consolidate the error checks and then to recover the rationales. Once the errors checks were consolidated, members began to systematically discuss rationales in the same order the checks are made during the run. If a rationale was not obtainable from some member of the group, a specific member was assigned the task of researching the rationale and recording his/her results on a form created for this purpose. On the form, the discipline specialist filled in the criteria information, what the check did and under what circumstances it was valid, and detailed the rationale of the criteria. A reference was also cited, those of the formal texts when possible and historical data if formal reference documents did not define the constraint or rationale. The historical data referred to previous 6-DOF SVDS output that was available from recent missions. This provided referential validation of the knowledge, both the constraints and their rationale (Reference 2). At a subsequent meeting the rationale was scrutinized by the group, aiding the group learning process and refining the information. The completed form was kept in a notebook for further reference and use.

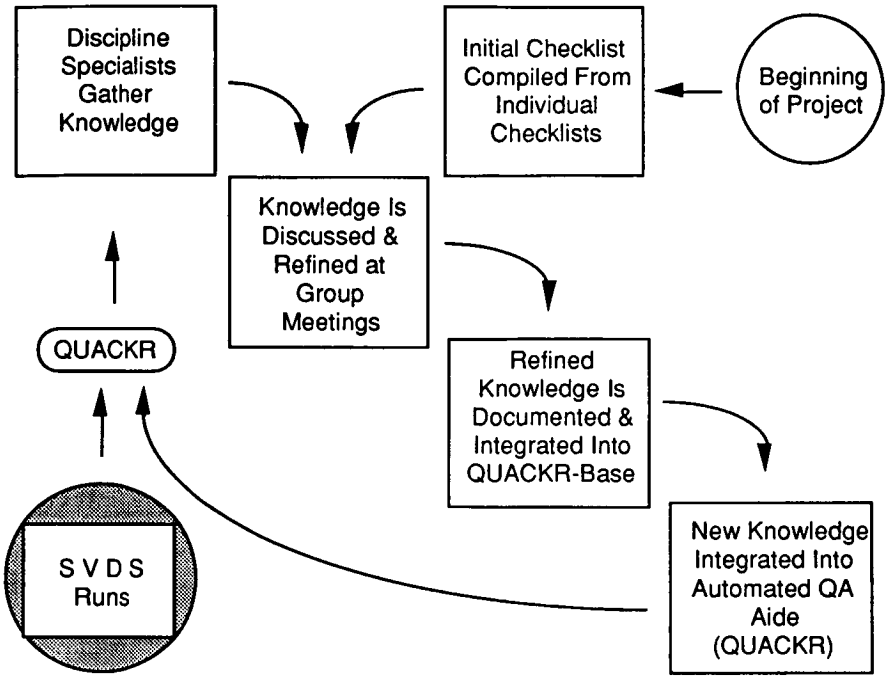
To support the group process and off-load time consuming acquisition of group expertise by the knowledge engineer, a project lead was recruited. The project lead's functions were to

- Evaluate the captured knowledge and determine deficiencies.
- Ensure the clarity of the recorded knowledge.
- Set the agenda for meetings.
- Hand out research assignments and follow-up to see that they are accomplished.
- Maintain a notebook that consolidates the captured knowledge.

Some initial training in group dynamics and knowledge capture was needed to prepare the project lead for these responsibilities. This training was provided by the knowledge engineer. After the process was underway, the knowledge engineer needed only to advise the project lead in these areas.

The approach used for the knowledge capture results in a self-perpetuating refinement process, Figure 1. As QUACKR (Quality Assurance Checker), the FORTRAN program is used, the error criteria and their rationale are placed under further scrutiny. In the course of their examinations, the discipline specialist gather more knowledge. Additions and improvements to the knowledge are brought into the group meetings by group members. The knowledge is refined during group discussions. When a consensus is reached and the information is comprehensible to the

knowledge engineer the subjective validation of the knowledge has also occurred (Reference 2). This improved knowledge is added to the knowledge-base, QUACKR-base, and used to upgrade QUACKR.



A Self-Sustaining Knowledge Capture Process

Figure 1

Interest in the knowledge capture process has been sparked by the potential of QUACKR. The member's of the Post-MECO group realize that the rationales are needed to validate QUACKR, and they were eager to carry out their assignments. Capture was completed in December, 1989, marking the end of Phase I. The rationales are accessible from the QUACKR-base to serve as off-line explanations to QUACKR. The documentation and accessibility of the QUACKR-base is an excellent improvement over the previous condition of the information. Through their experiences, the group has discovered that the rationales also give a good head start toward finding a resolution.

Along with improving the knowledge in QUACKR, the individuals in the group have improved their own knowledge at an accelerated pace. Discussing the knowledge at meetings not only refines the knowledge, but also passes the knowledge on to other members of the group. The group also has gained "meta-knowledge": they know more about what they know. Individuals had developed areas of specialization and these specialists became identifiable during the group meetings.

McGraw and Harbison-Briggs (Reference 3) emphasize the importance of maintaining good, working relationships between the knowledge engineer and the group and between members of the group. The project lead has established working relationships with the group which are extended to the knowledge capture context. The project lead is also familiar with the relationships among the experts. This has lessened the knowledge engineer's concern over these relationships.

KNOWLEDGE TOOL

As a post processor to the SVDS run, an expert system, QUACKR (Quality Assurance Checker), which checks boundary constraints was built. It is a FORTRAN program that uses the defined and verified list of boundary constraints to detect violations and print warning messages.

QUACKR is the most obvious physical results of the knowledge capture process. Most important about this knowledge-based aid (KBA), even though it is conventionally implemented, is that it provides consistent, rigorous performance of the fault detection task. QUACKR began with approximately 300 checks accumulated from the group's checklists. The group knowledge capture process has resulted in the addition of over 100 checks. The expert's schedules do not allow them to perform the 400 time-consuming checks manually. QUACKR performs the evaluations in three minutes of UNIVAC 1192 computer time. As the knowledge in QUACKR is upgraded, the time savings will increase. As the flight rate increases, the use of QUACKR will increase, thereby increasing the cost savings QUACKR provides.

THE KNOWLEDGE-BASE MANAGEMENT SYSTEM

The electronic KBMS, written in HyperCard, will eventually replace the project leader's notebook as the place to maintain and access the knowledge. The knowledge-base is referred to as the QUACKR-base. Eventually the QUACKR-base will include all error detection constraints and their rationale (Phase I), causes of constraint violations and resolutions (Phase II). The QUACKR-base provides a user-friendly means for the group to access the knowledge. This allows the group to

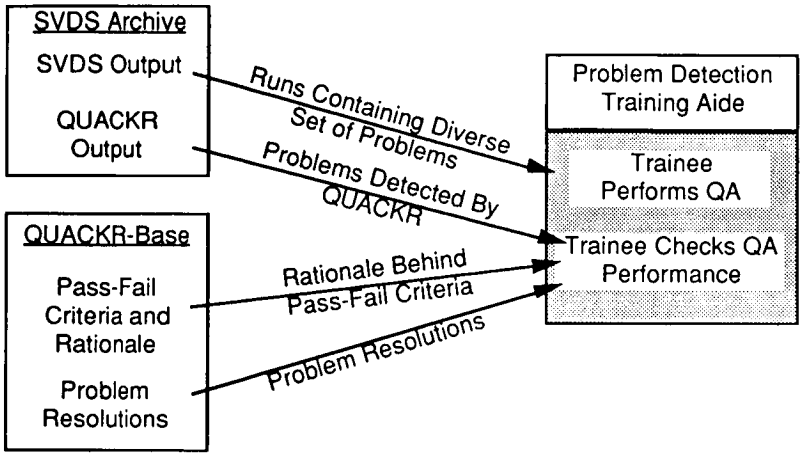
- Simultaneously maintain Quackr and Quackr-base
- Capture knowledge with currently undetermined structure
- Support changes to knowledge easily
- Remain cognizant of the knowledge utilized by QUACKR, preventing it from becoming a "black box".
- Support new hire training.
- Reference information pertinent to problem resolution.

The knowledge is stored in the knowledge-base in a format which is much more readable than FORTRAN code. Keeping the QUACKR-base separate from QUACKR would introduce a point in the process where the knowledge in QUACKR and that in the QUACKR-base can become mismatched. The code generation capability prevents this from occurring. This utility allows the user to generate new QUACKR code whenever refinements are made to the knowledge in the QUACKR-base, keeping the knowledge in QUACKR and the QUACKR-base consistent. The validity of the knowledge in the knowledge-base, therefore, reflects the validity of the knowledge in the KBA. Verification of the knowledge in the KBMS, both QUACKR and QUACKR-base, is made simpler through the use of consistency and completeness routines written in HyperCard's scripting language.

Another reason for choosing the HyperCard platform is that it allows a developer to create different structures for the various types of knowledge. This flexibility allows additional knowledge types to be included in the knowledge-base and allows changes in the current structure of knowledge. For example, parameter knowledge is stored as objects and checks are stored as if-then statements. Since the capture of problem resolution knowledge has not begun yet, the appropriate structure for storing that knowledge is unknown. However, as mentioned previously, the rationale behind a check often points to resolutions should that check fail. Consequently, some problem resolutions have been captured. Currently, these are stored in variable length text fields. After more are captured, an appropriate structure will be decided upon, and the problem resolutions will be stored in the QUACKR-base along with the problem detection knowledge. To exemplify how the knowledge may change, as the novices in the group become experts, their method of problem resolution may become more schema-driven than data-driven (Reference 1). The flexibility of the KBMS will also allow extensions to restructure the knowledge.

As shown, the KBMS is easily extended to contain additions to and changes in the knowledge. The new and refined knowledge can in turn be used to develop other automated aids, i.e., a rule-based problem resolution expert system, which will be managed using the KBMS. The hyperlink capability of HyperCard allows links to be built between the types of knowledge so that logical connections between the types are maintained. In this way the hyperlinks also define the relationships between the knowledge tools.

Since the QUACKR-base contains the relevant knowledge it stands alone as an excellent reference source. The QUACKR-base is not readily available to the users currently, so, it is more convenient to parse the knowledge-base and create a reference document. The KBMS has report utilities which will do this. The knowledge in the QUACKR-base will also benefit training. Presently, trainees learn experientially. Various scenarios are being considered which utilize the knowledge in the QUACKR-base for training, such as the one in Figure 2.

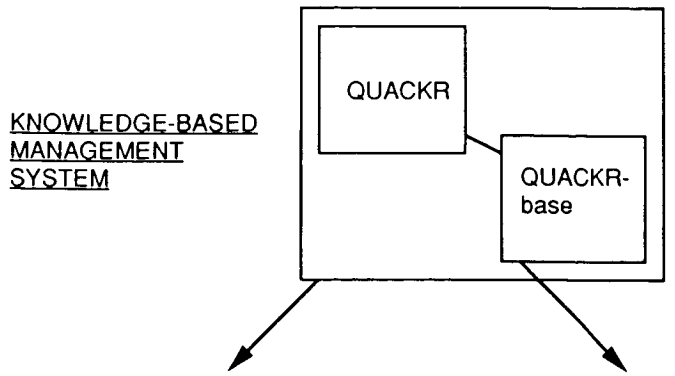


A Scenario For Training Using The QUACKR-Base

Figure 2

This training scenario involves compiling a set of SVDS runs with a diverse set of problems and having the trainee detect faults manually. The trainee could check his/her own performance against the QUACKR output. The trainee could also resolve any problems manually and match his/her performance against resolutions in the QUACKR-base. This scenario would provide more in depth training, with less demand on experienced personnel, in a shorter amount of time than the present method of training. By training with the knowledge in the knowledge-base, the group remains aware of what knowledge is in QUACKR and the rationale behind the QA process, which keeps QUACKR from becoming a "black box."

The KBMS satisfies many objectives of the project with its capabilities which manage the acquired QA knowledge and the knowledge tools. Figure 3 depicts the functions and capabilities of the KBMS.



Management functions	Capabilities
• Autonomous management of QUACKR	Automatic code generation
• Support long-term continuous capture	Flexible knowledge structuring
• Easily change QA knowledge when necessary	User friendly platform
• Support knowledge retention	Contains the QA knowledge and provides an easy reference source
• Provide QA knowledge documentation	Report generation of QA knowledge
• Support training	Reference source and documentation

KBMS Functions and Capabilities

Figure 3

CONCLUSIONS

It is possible for the Post-MECO group to sustain their own knowledge capture process and maintain the knowledge-based management system themselves. Two things lead us to believe this is possible. One is the project lead. Once the knowledge capture process is started, the knowledge engineer becomes a consultant to the project lead. The Post-MECO group is already capturing their knowledge self-sufficiently. Maintaining the QUACKR-base should be easy enough that anyone can do the maintenance. However, only one person from the group will be trained as a QUACKR-base manager and allowed to update the QUACKR-base knowledge for configuration control reasons. The knowledge engineer will remain in charge of upgrading the QUACKR-base software when needed. With the consistent effort of the QUACKR-base manager to update the QUACKR-base with current knowledge, the automatic effortless export of knowledge from the QUACKR-base into QUACKR will satisfy the maintenance requirement of the knowledge-based system.

This self-sustaining knowledge capture process and resulting KBMS fulfills the objectives of this task. Through the approach chosen the dispersed rationales were captured while supporting the in-progress training of the engineers. The rationale refinement meetings were within the discipline specialists time constraints and were able to increase the level of knowledge within the group. With the help of the project lead the knowledge engineers were given some relief from the project.

The reasons for doing this task were accomplished with the knowledge tools. The knowledge tools reduced the QA task time sufficiently and also documented the available and acquired QA knowledge. The availability of this knowledge to an engineer should prevent QUACKR from becoming a black box.

The management of this knowledge is provided by the capabilities of the KBMS, QUACKR and QUACKR-base. Long-term continuous capture can be maintained from the continuous generation of QUACKR output and the flexibility of the knowledge structure within QUACKR-base. Changes to the knowledge are easily accommodated because an easy to use system was chosen, Hypercard. Knowledge retention and training are supported with report generation and the accessibility to a user friendly reference source, the QUACKR-base.

Because the prototype project has been successful in attaining the time reduction of the QA task and has also been able to meet many other objectives this process has begun in two other areas within FD&DD. These new groups have begun with the compiling of a master checklist and capturing rationale. This second effort will be based on what was learned during the Post-MECO QUACKR project and the unique needs of this group. There are at least ten areas within FD&DD that can benefit from the type of knowledge capture process and resulting knowledge-based systems that have evolved from the Post-MECO experience. Knowledge-based systems are planned to be in place by 1992 to support the increasing flight rate.

REFERENCES

1. LaFrance, Marianne, "The Quality of Expertise: Implications of Expert-Novice Differences for Knowledge Acquisition". A CM Newsletter, Special Issue On: Knowledge Acquisition, No. 108, April 1989.

2. Boose, John and Shaw, Mildred. "IJCAI-89 Tutorial Notes: Knowledge Acquisition for Knowledge-Based Systems," Detroit, Michigan, 1989.

3. McGraw, Karen and Harbison-Briggs, Karan. Knowledge Acquisition Principles and Guidelines. Prentice Hall, New Jersey, 1989.